Claims

1   1. A method of analyzing program execution within an

2 operating system of a multithreaded environment, comprising:

3   accumulating diagnostic data pertaining to a thread accessing a

4 resource, the execution of a thread being predicated upon the thread's access to

5 the resource;

6   storing the diagnostic data within a data structure at a location

7 in the data structure correlated to the resource.

1   2. The method according to claim 1, wherein the diagnostic

2 data includes data selected from a group consisting of: a time measurement,

3 program code executed by the thread, an invocation stack, pointer data and

4 some combination, thereof.

1   3. The method according to claim 1, wherein the data structure

2 comprises a hash bucket.

1   4. The method according to claim 1, further comprising

2 determining the resource.

1   5. The method according to claim 4, wherein determining the

2 resource includes reading contents of a task dispatcher.

1   6. The method according to claim 1, further comprising storing

2   information identifying the resource.

1   7. The method according to claim 1, further comprising

2   matching an identifier corresponding to the resource to a correlative identifier

3   corresponding to the data structure.

1   8. The method according to claim 7, further comprising

2   reassigning the identifier to a second resource.

1   9. The method according to claim 7, further comprising

2   assigning the correlative identifier to the data structure.

1   10. The method according to claim 1, further comprising

2   detecting a locking occurrence.

1   11. The method according to claim 10, further comprising

2   calculating a time increment corresponding to a duration that the thread

3   remains locked.

1   12. The method according to claim 11, further comprising

2   storing the time increment within the data structure.

1   13. The method according to claim 10, further comprising

2   recording the time corresponding to the locking occurrence.

1   14. The method according to claim 1, further comprising

2   detecting a removal of the lock.

1   15. The method according to claim 14, further comprising

2   recording a time instance corresponding to the removal of the lock.

1   16. The method according to claim 10, further comprising

2   recording program data relating to code executed by the thread prior to the

3   locking occurrence.

1   17. The method according to claim 16, further comprising

2   retrieving the program data from an invocation stack.

1   18. The method according to claim 1, further comprising

2   displaying the diagnostic data.

1   19. A method of analyzing program execution within a

2 computer system having a plurality of threads accessing a plurality of

3 resources, comprising:

4   calculating a time increment reflective of a duration a thread of

5 the plurality of threads waits for access to a resource of the plurality of

6 resources, the execution of the thread being predicated upon the thread's

7 access to the resource; and

8   storing the time increment within a bucket of a plurality of

9 buckets comprising a hash array, each bucket being correlated to the resource.

1   20. The method according to claim 19, further comprising

2 reallocating the plurality of resources to the plurality of buckets to group the

3 diagnostic data with a different scheme.

1   21. An apparatus comprising:

2   at least one processor configured to execute a plurality of

3 threads;

4   a memory; and

5   program code resident in the memory and configured to execute

6 on the at least one processor, the program code configured to accumulate

7 diagnostic data pertaining to a thread accessing a resource, the execution of a

8 thread being predicated upon the thread's access to the resource, and to store

9 the diagnostic data within a data structure at a location in the data structure

10 correlated to the resource.

1   22. The apparatus according to claim 21, wherein the

2 diagnostic data includes data selected from a group consisting of: a time

3 measurement, program code executed by the thread, an invocation stack,

4 pointer data and some combination, thereof.

1   23. The apparatus according to claim 21, wherein the lock of

2 memory comprises a hash bucket.

1   24. The apparatus according to claim 21, wherein the program

2 code initiates a determination of the resource.

1   25. The apparatus according to claim 21, wherein the program

2   code initiates storing information identifying the resource.

1   26. The apparatus according to claim 21, further comprising

2   matching an identifier corresponding to the resource to a correlative identifier

3   corresponding to the data structure.

1   27. The apparatus according to claim 26, wherein the program

2   code initiates reassigning the identifier to a second resource.

1   28. The apparatus according to claim 26, wherein the program

2   code initiates assigning the correlative identifier to the data structure.

1   29. The apparatus according to claim 21, wherein the program

2   code initiates a detection of a locking occurrence.

1   30. The apparatus according to claim 21, wherein the program

2   code initiates a calculation of a time increment corresponding to a duration

3   that the thread remains locked.

1   31. The apparatus according to claim 30, wherein the program

2   code initiates storing the time increment within the data structure.

1   32. The apparatus according to claim 21, wherein the program

2   code initiates recording a time corresponding to a locking occurrence.

1   33. The apparatus according to claim 21, wherein the program

2   code initiates detecting a removal of the lock.

1   34. The apparatus according to claim 33, wherein the program

2   code initiates recording a time instance corresponding to the removal of the

3   lock.

1   35. The apparatus according to claim 29, wherein the program

2   code initiates recording program data relating to code executed by the thread

3   prior to a locking occurrence.

1   36. The apparatus according to claim 35, wherein the program

2   code initiates retrieval of the program data from an invocation stack.

1   37. The apparatus according to claim 21, wherein the program

2   code initiates a display of the diagnostic data.

1   38. A program product, comprising:

2   program code for analyzing program execution within an

3   operating system of a multithreaded environment, wherein the program code is

1 configured to accumulate diagnostic data pertaining to a thread accessing a

2 resource, the execution of a thread being predicated upon the thread's access to

3 the resource, and to store the diagnostic data within a block of the memory

4 correlated to the resource; and

5 a signal bearing medium bearing the program code.

1 39. The program product of claim 38, wherein the signal

2 bearing medium includes at least one of a recordable medium and a

3 transmission-type medium.

4